ระบบค้นหากองทุนที่เหมาะสมกับผู้ลงทุน

Mutual Fund Searching System

คณะผู้จัดทำ

นาย ฐปนพงศ์ จันตะมะ 600510542

นาย วชิระ นรสิงห์ 600510576

อาจารย์ที่ปรึกษา

ผู้ช่วยศาสตราจารย์ ดร.อารีรัตน์ ตรงรัศมีทอง

รายงานนี้เป็นส่วนหนึ่งของวิชา การออกแบบและพัฒนาออนโทโลยี (204424)

ภาคการศึกษาที่ 2 ปีการศึกษา 2563

# คำนำ

รายงานเล่มนี้จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของวิชา การออกแบบและพัฒนาออนโทโลยี (204424) โดยคณะผู้จัดทำได้สร้างระบบสำหรับค้นหากองทุนให้เหมาะสมกับผู้ลงทุน โดยใช้ฐานข้อมูลในรูปแบบ ออนโทโลยีซึ่งมีความสามารถสำหรับจัดเก็บฐานข้อมูลในรูปแบบฐานข้อมูลแบบความรู้

โดยคณะผู้จัดทำหวังว่าการจัดทำเอกสารฉบับนี้จะช่วยให้ข้อมูลที่เป็นประโยชน์ต่อผู้สนใจในการออกแบบฐานข้อมูลแบบ ออนโทโลยี และ การใช้งานเครื่องมือสำหรับใช้ฐานข้อมูลแบบ ออนโทโลยี

นาย ฐปนพงศ์ จันตะมะ

นาย วชิระ นรสิงห์

# สารบัญ

**เรื่อง หน้า**

[คำนำ a](#_Toc66969625)

[สารบัญ b](#_Toc66969626)

[บทที่ 1 บทนำ 1](#_Toc66969627)

[1.1 แนวคิดหลัก (Concept) 1](#_Toc66969628)

[1.2 วัตถุประสงค์ 1](#_Toc66969629)

[1.3 ข้อความที่เกี่ยวข้อง 1](#_Toc66969630)

[บทที่ 2 เครื่องมือ และ ทฤษฎีที่เกี่ยวข้อง 3](#_Toc66969631)

[2.1 เครื่องมือที่ใช้ในการพัฒนาระบบ 3](#_Toc66969632)

[2.1.1 Related Technologies 3](#_Toc66969633)

[2.2 ภาษาที่ใช้พัฒนาระบบและภาษาที่ใช้ในการ Query Ontology 3](#_Toc66969634)

[บทที่ 3 การออกแบบ 5](#_Toc66969635)

[3.1 Ontology Schema 5](#_Toc66969636)

[3.2 Class Hierarchy 7](#_Toc66969637)

[3.3 Object Properties 11](#_Toc66969638)

[3.4 Data Properties 11](#_Toc66969639)

[3.5 ตัวอย่างของ Instance 12](#_Toc66969640)

[3.6 Relation between objects 14](#_Toc66969641)

[3.7 ส่วนก่อประสาน (User Interface) 15](#_Toc66969642)

[3.8 สถาปัตยกรรมของระบบ (Software Architecture) 16](#_Toc66969643)

# บทที่ 1 บทนำ

## แนวคิดหลัก (Concept)

ระบบค้นหากองทุนที่เหมาะสมกับนักลงทุน เนื่องจากการลงทุนเป็นเรื่องที่จำเป็นต้องใช้ความรู้ในการพิจารณา ทำให้นักลงทุนจะต้องศึกษาข้อมูลจำนวนมากก่อนตัดสินใจ แต่ระบบที่จัดทำจะนำ Knowledge Base มาช่วยในการตัดสินใจและเลือกกองทุนที่เหมาะสมกับนักลงทุน และ แสดงข้อมูลการคาดการณ์ ผลตอบแทนที่ได้ผ่านระบบภายนอก โดยใช้การเขียนโปรแกรมสำหรับแสดงผลคาดการณ์ในอนาคตทั้งนี้ผลคาดการณ์เป็นเพียงการนำข้อมูลจากอดีตมาประมวลผล

## วัตถุประสงค์

เพื่อแสดงกองทุนที่เหมาะสมกับ เป้าหมาย, เงินทุน และ ระยะเวลาการลงทุน ของผู้ลงทุน

## ข้อความที่เกี่ยวข้อง

* บริษัทหลักทรัพย์จัดการกองทุน
* นักลงทุน
* ราคา
  + NAV
  + ราคาซื้อ
  + ราคาขาย
* กองทุนรวม
  + กองทุนเปิด
  + กองทุนปิด
  + กองทุนรวมตลาดเงิน
  + กองทุนรวมตราสารหนี้
  + กองทุนรวมตราสารทุน
  + กองทุนรวมผสม
  + กองทุนรวมทรัพย์สินทางเลือก
  + กองทุนรวมต่างประเทศ
* ความเสี่ยง
  + เสี่ยงน้อย (1)
  + เสี่ยงมาก (8)
* สินทรัพย์
  + กองทุน
  + หุ้น
  + ตราสารหนี้ภาครัฐ
  + หุ้นกู้
  + ทองคำ
  + น้ำมัน
  + อสังหาริมทรัพย์
* ผลตอบแทน
  + ปันผล
  + สะสมมูลค่า

# บทที่ 2 เครื่องมือ และ ทฤษฎีที่เกี่ยวข้อง

## เครื่องมือที่ใช้ในการพัฒนาระบบ

## 2.1.1 Related Technologies

* + - * React Framework สำหรับสร้าง Client-application
      * Express.js (a server framework)
      * Prisma (an ORM for creating Object-oriented model for Relational Database)
      * Fast API (a server framework for python)
      * Axios (Library for creating Http Request to RESTful API)
      * PostgreSQL สำหรับเก็บข้อมูลที่จำเป็นต้อง Update อยู่ตลอด

**ยกตัวอย่างเช่น** NAV, ราคาซื้อ, ราคาขาย, รายละเอียดกองทุน

* + - * Protégé (a knowledge base administration tool) สำหรับสร้าง Ontology Schema
      * Apache Jena Fuseki สำหรับ Ontology Server เพื่อรับ Query และ Update Command
      * fbProphet สำหรับทำนายอัตราการเติบโตของกองทุน

**2.1.2 Outer API**

* + - * SEC API (กลต.) สำหรับเรียก Fact Sheet และ NAV ของกองทุนรวมทั้งหมด

## ภาษาที่ใช้พัฒนาระบบและภาษาที่ใช้ในการ Query Ontology

**2.2.1** **Client App และ Server**

* + - * TypeScript
      * Python
      * SQL Language

**2.2.2 Ontology Server (Jena Fuseki)**

* + - * SPARQL Command
      * Turtle Type extension

# บทที่ 3 การออกแบบ

## Ontology Schema

การออกแบบ Ontology Schema ในระบบค้นหากองทุนนั้นเริ่มจากการออกแบบที่ตัวกองทุนซึ่งสำคัญที่สุดโดยกองทุนมี Sub-Class ที่ได้มากจากการจำแนกของ กลต. จากเอกสาร นิยามประกาศ สน.87/2558 ภาคผนวก 2 โดยออกแบบไว้ดังนี้

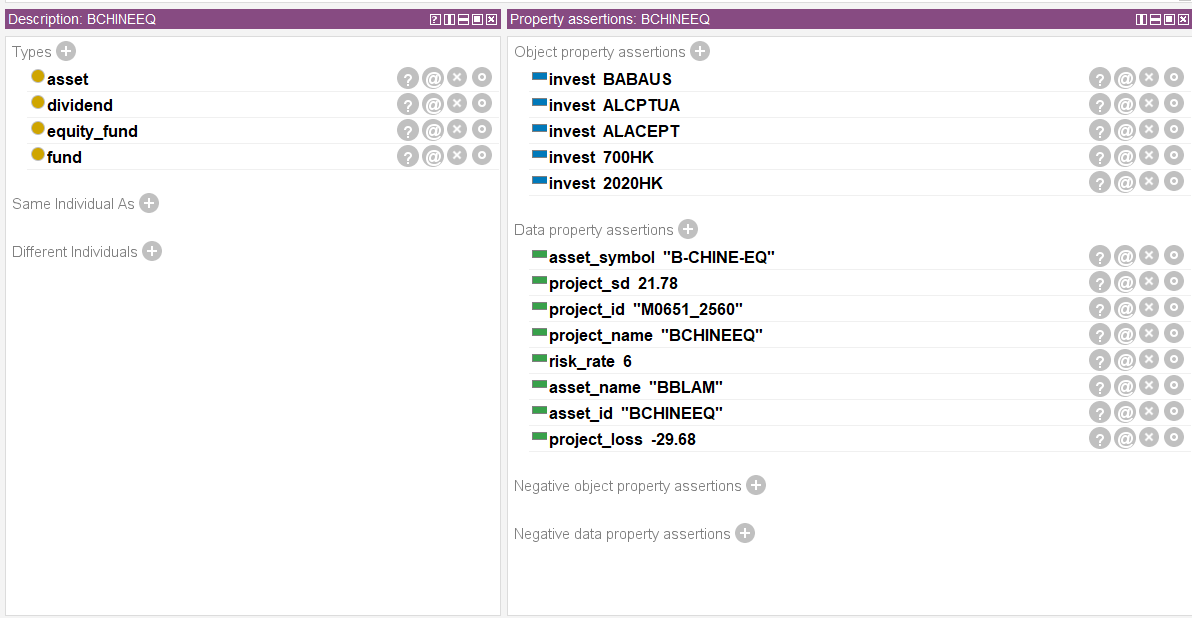
* **กองทุน**
  + **กองทุนที่จ่ายปันผล**
  + **กองทุนตราสารหนี้**
  + **กองทุนตราสารทุน**
  + **กองทุนผสม**
  + **กองทุนทางเลือก**
  + **อื่น**

นอกจากนี้กองทุนยังมี Data-Properties หรือ ข้อมูลภายใน Object โดยการเลือก Data-Properties ทำได้จากการวิเคราะห์การเลือกกองทุนเบื้องต้นซึ่งประกอบไปด้วย

* ชื่อรหัสกองทุน
* ผลตอบแทนที่ทำได้ ตั้งแต่เริ่มจัดตั้ง
* ระดับความเสี่ยง
* การขาดทุนมากที่สุด ตั้งแต่จัดตั้ง
* ค่าความคลาดเคลื่อน

เนื่องจากข้อมูลเหล่านี้มีการเปลี่ยนแปลงเพียงปีละครั้งจึงเหมาะสมที่จะนำเข้าไปไว้ใน Knowledge Base เพื่อทำการวิเคราะห์หากองทุนที่เหมาะสม

**ตัวอย่าง Individual ของ กองทุนรวม**



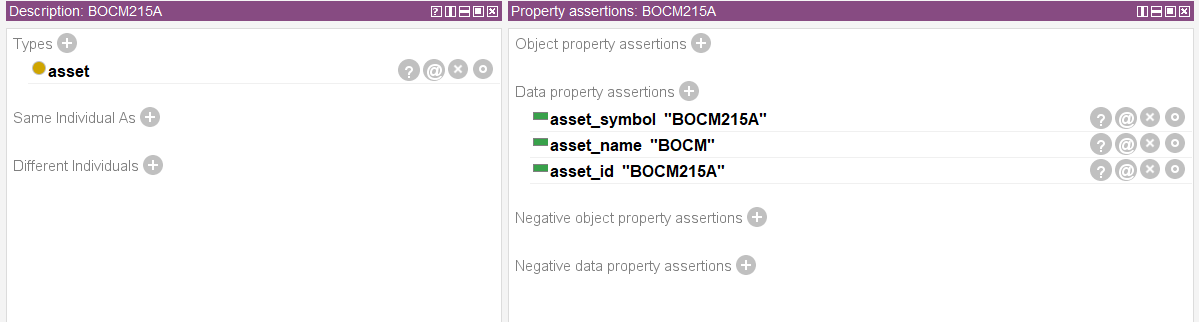
ในส่วนต่อไปคือส่วนของ สินทรัพย์ที่กองทุนลงทุนโดยจะถูกเชื่อมความสัมพันธ์กับกองทุนโดย Object Properties ที่ชื่อว่า Invest โดยลักษณะของ Class สินทรัพย์มีดังนี้

* สินทรัพย์
  + เงินฝาก
  + หน่วยลงทุน
  + หุ้น
  + ตราสารหนี้
  + หุ้นกู้
  + ทองคำ
  + ใบสำแดงสิทธิ

โดย สินทรัพย์จะมี Data Properties เพียง 3 เท่านั้นซึ่งประกอบไปด้วย

* รหัสของสินทรัพย์
* ชื่อของสินทรัพย์
* สัญลักษณ์ของสินทรัพย์

**ตัวอย่าง Individual ของ สินทรัพย์**



**ตัวอย่างการเรียก Query Command และ ผลลัพธ์สำหรับเรียกกองทุนใน Ontology**

**Parameter: dividend: false, loss: 5**

const command = `

  SELECT ?project\_id ?name ?draw\_down ?profit ?sd ?risk (?profit+?sd as ?max\_profit)

(?profit-?sd as ?min\_profit)

  WHERE{

    ?fund mat:project\_id ?project\_id;

          mat:project\_name ?name;

          mat:project\_loss ?draw\_down;

          mat:project\_profit ?profit;

          mat:project\_sd ?sd;

          mat:risk\_rate ?risk;

          ${dividend ? 'rdf:type mat:dividend;' : ''}

          FILTER ( ?draw\_down >= ${loss} && (?sd + ?profit >= ${profit} || ?profit - ?sd >= 0))

          ${risk? `FILTER (?risk >= ${riskValue[risk].start}

&& ?risk <= ${riskValue[risk].end})`: ''}

  }

  ORDER BY DESC(?profit+?sd) DESC(?profit-?sd) DESC(?draw\_down)

  ${litmit ? `LIMIT ${litmit}` : ''}

  `

**ผลลัพธ์จาก Query Command**

[

    {

        "project\_id": "M0512\_2563",

        "name": "KCHINARMF",

        "draw\_down": "-2.16",

        "profit": "8.02",

        "sd": "17.19",

        "risk": "6",

        "max\_profit": "25.21",

        "min\_profit": "-9.17"

    },

    {

        "project\_id": "M0110\_2563",

        "name": "BCHINAARMF",

        "draw\_down": "-2.48",

        "profit": "8.05",

        "sd": "14.21",

        "risk": "6",

        "max\_profit": "22.26",

        "min\_profit": "-6.16"

    },

    {

        "project\_id": "M0470\_2563",

        "name": "KUSARMF",

        "draw\_down": "-2.73",

        "profit": "5.24",

        "sd": "16.15",

        "risk": "6",

        "max\_profit": "21.39",

        "min\_profit": "-10.91"

    },

    {

        "project\_id": "M0108\_2563",

        "name": "BMAPS100",

        "draw\_down": "-1.81",

        "profit": "4.89",

        "sd": "6.4",

        "risk": "6",

        "max\_profit": "11.29",

        "min\_profit": "-1.51"

    },

    {

        "project\_id": "M0534\_2563",

        "name": "SCBGEESG",

        "draw\_down": "-1.08",

        "profit": "5.37",

        "sd": "2.29",

        "risk": "6",

        "max\_profit": "7.66",

        "min\_profit": "3.08"

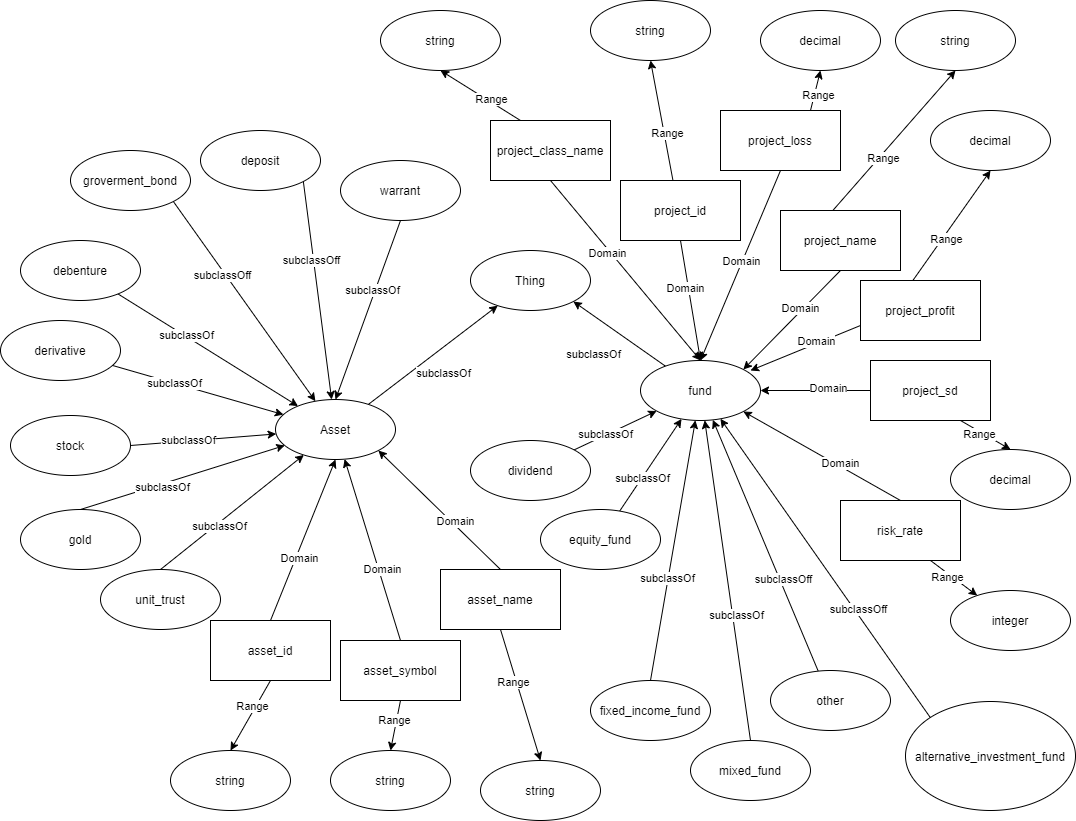
    }

]

## Class Hierarchy

Class Hierarchy คือ แผนภาพแสดงความสัมพันธ์ระหว่าง Class และ Property ซึ่งมีความสัมพันธ์ในลักษณะ Subject => Predicate => Object โดยใช้สัญลักษณ์ในการแสดงแผนภาพดังนี้

|  |  |
| --- | --- |
| **Symbol** | **Meaning** |
|  | แสดงถึง Class ที่เป็น Subject |
| Relation | แสดงถึง ความสัมพันธ์ระหว่าง Class ถึง Class หรือ Class ถึง Property |
| Property | แสดงถึงความหมายของความสัมพันธ์ที่เกิดขึ้นระหว่าง Class โดยผ่าน Relation |



แผนภาพแสดง Class Hierarchy

## Object Properties

|  |  |  |
| --- | --- | --- |
| **Name** | **Domain** | **Range** |
| invest | Fund | Asset |

## Data Properties

|  |  |  |
| --- | --- | --- |
| **Name** | **Domain** | **Range** |
| asset\_id | asset | string |
| asset\_name | asset | string |
| asset\_symbol | asset | string |
| project\_class\_name | fund | string |
| project\_id | fund | string |
| project\_loss | fund | decimal |
| project\_name | fund | string |
| project\_profit | fund | decimal |
| project\_sd | fund | decimal |
| risk\_rate | fund | integer |

## ตัวอย่างของ Instance

1. **Class:** fund

**Sub-Class**: equity-fund

**Instance Name:** SCBSET50

**Data Property**

|  |  |  |
| --- | --- | --- |
| **Subject** | **Type** | **Value** |
| project\_id | string | M0415\_2562 |
| loss | float | 36.5% |
| risk\_rate | int | 6 |
| sd | float | 18.76% |
| profit | float | 5.04% |

**Object Property**

|  |  |  |
| --- | --- | --- |
| **Subject** | **Predicate** | **Object** |
| SCBSET50 | invest | PTT |
| SCBSET50 | invest | AOT |
| SCBSET50 | invest | CPALL |
| SCBSET50 | invest | DELTA |
| SCBSET50 | invest | AIS |

1. **Class:** asset

**Sub-class:** stock

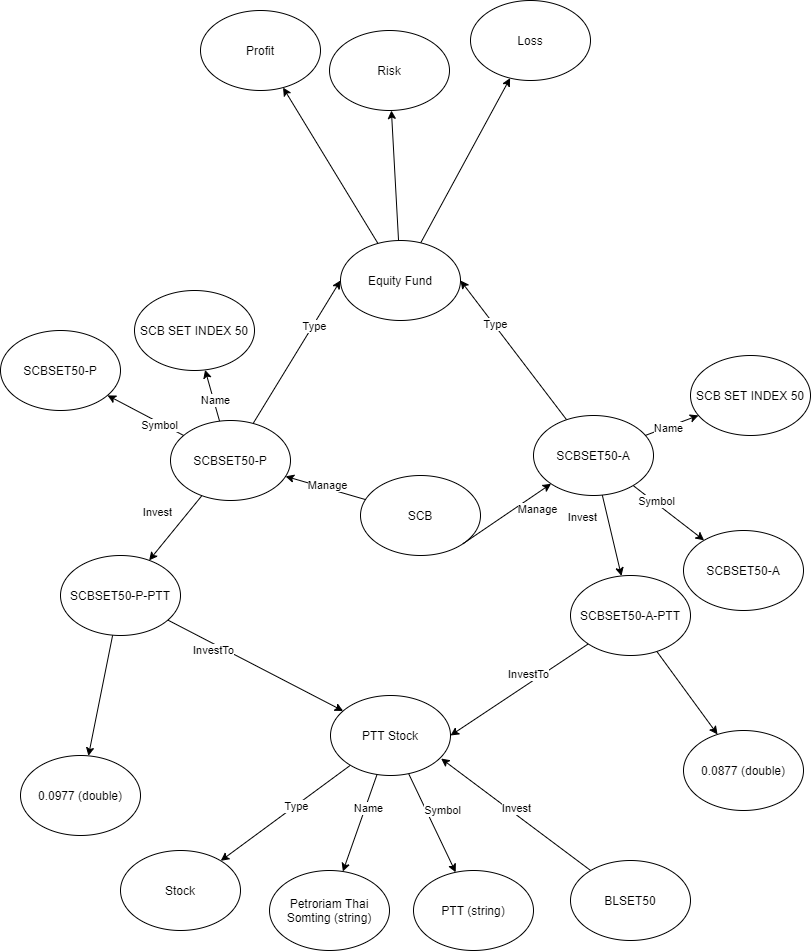
**Instance Name:** AOT

**Data Property**

|  |  |  |
| --- | --- | --- |
| **Property** | **Type** | **Value** |
| asset\_id | string | AOT |
| asset\_name | string | AIRPORTS OF THAILAND PUBLIC COMPANY LIMITED |
| Asset\_symbol | string | AOT |

## Relation between objects

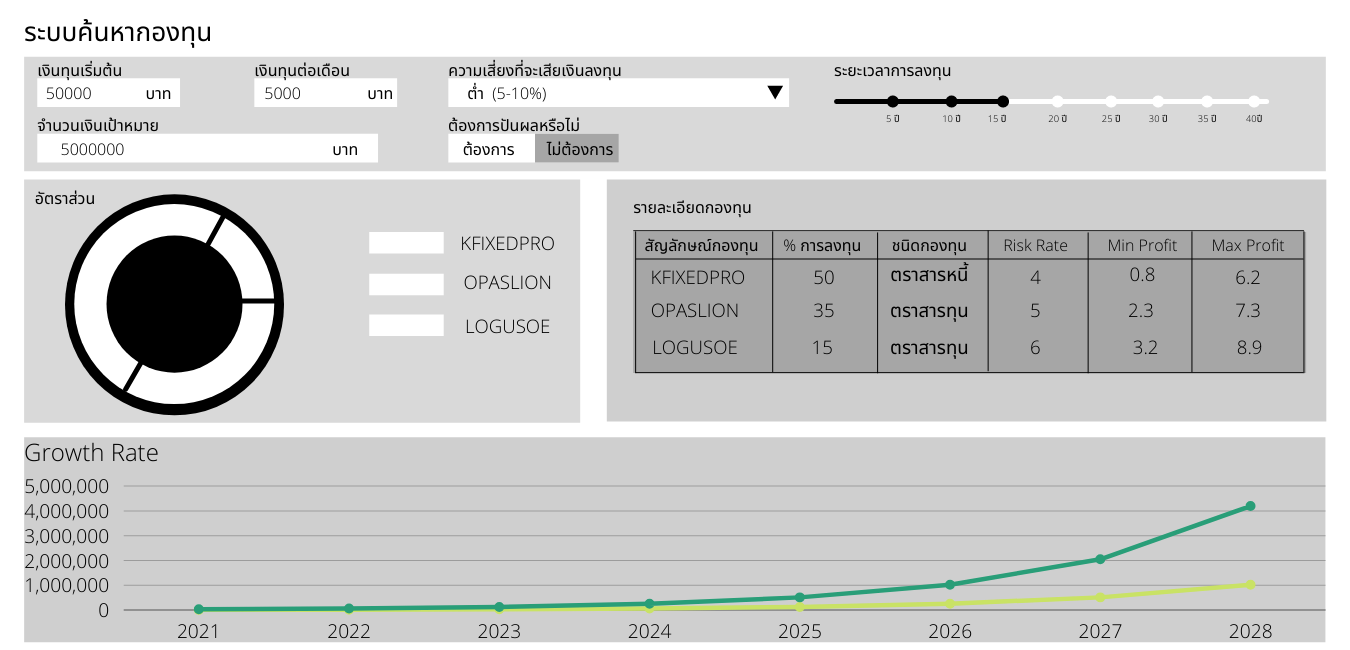
ตัวอย่างของ Individual ที่ถูกเชื่อมความสัมพันธ์เข้ากับทั้ง Object Property และ Data Property



## ส่วนก่อประสาน (User Interface)

การออกแบบส่วนก่อประสานประกอบไปด้วย 5 ส่วนโดยประกอบไปด้วย

1. ส่วนรับเข้าข้อมูล
   1. เงินทุนเริ่มต้น
   2. เงินทุนต่อเดือน
   3. ความเสี่ยงที่รับได้
   4. ระยะเวลาการลงทุน
   5. จำนวนเงินเป้าหมาย
   6. การจ่ายปันผล
   7. ปุ่มค้นหา
2. ส่วนแสดงอัตราส่วนการลงทุน
3. ส่วนแสดงรายละเอียดของกองทุนที่ถูกเลือก
4. ส่วนแสดงการคาดการณ์มูลค่ากองทุนทั้งหมดในอนาคต
5. ส่วนแสดงผลลัพธ์ความสามารถในการทำตามเป้าหมาย
   1. เส้นที่ 1 แสดงมูลค่ากองทุนทั้งหมด
   2. เส้นที่ 2 แสดงจำนวนเงินต้นทั้งหมด

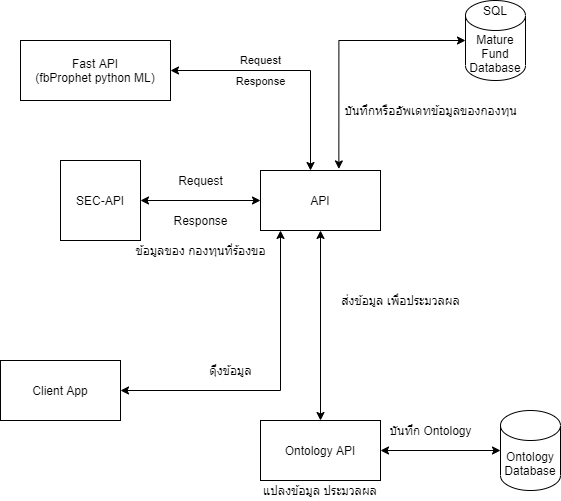


## สถาปัตยกรรมของระบบ (Software Architecture)

การออกแบบสถาปัตยกรรมของระบบโดยออกแบบในลักษณะ Three-Tier Architecture โดยมี Application Layer แยกจาก Database Layer และมี Presentation Layer โดยจำแนกได้ดังนี้

1. Presentation Layer
   1. Client-App
2. Application Layer
   1. Mutual-Fund-Ranking API: เป็น Entry Point สำหรับ Client-App เพื่อเรียกใช้ API อื่นๆ
   2. SEC-API: สำหรับ ข้อมูลของกองทุนรวม จาก กลต.
   3. Predication API: สำหรับ ประมวลผล และ สร้างการคาดการณ์กองทุนผ่าน fbPhophet
   4. Ontology API: สำหรับ ประมวลและเรียกข้อมูลภายใน Knowledge Base
3. Database Layer
   1. Turtle Extension: Ontology Schema
   2. PostgreSQL: Relational Database

โดยภาพรวมของสถาปัตยกรรมทั้งหมดเป็นตาม รูปภาพด้านล่าง



## ทฤษฎีที่เกี่ยวข้อง (Related Theory)

ในหัวข้อนี้กล่าวถึงกระบวน และ ทฤษฎีทั้งหมดที่ใช้ภายในระบบจัดการกองทุน

1. **หลักการคำนวณผลตอบแทนสูงสุดและมากที่สุดของ กองทุนรวม ในระยะเวลา 1 ปี**

ในการหาผลตอบแทนสูงสุดของกองทุนรวมนั้นจำเป็นต้องมีข้อมูลประกอบดังนี้

* + - ผลตอบแทนตั้งแต่จัดตั้งกองทุน (initProfit)
    - ค่าความผันผวนตั้งแต่จัดตั้งกองทุน (S.D.)

โดยสามารถคำนวณจากสมการดังนี้

1. **ความเสี่ยงและผลตอบแทน (Risk And Reward)**

Risk and Reward คือ การนำความเสี่ยงมาเทียบกับผลตอบแทนที่เราจะได้รับนั้นคุ้มค่าหรือไม่ โดยใช้วิธีง่ายๆ โดยนำความเสี่ยงหรือการขาดทุนมานับให้คุ้มค่าต่อการลงทุนยกตัวอย่างเช่นหากขาดทุนที่

5 % ของเงินทุน ควรที่จะได้ผลตอบแทนมากกว่า 5 % เพื่อที่จะได้รับผลตอบแทนที่คุ้มค่ากับความเสี่ยงที่ให้กับการลงทุนแต่ละครั้ง โดยใน Project ให้ Risk: Reward อยู่ที 1:1.25